CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | | |

Internal Assesment Test –I

| Sub: | MICROPROCESSOR | | | | | | | Code: | 10EC62 |
|---|---|---|---|---|---|---|---|---|---|
| Date: | 27 / 03 / 2017 | Duration: | 90 mins | Max Marks: | 50 | Sem: | VI | Branch: | ECE/TCE |

Answer Any FIVE FULL Questions

| | | Marks | ..... CO | RBT |
|---|---|---|---|---|
| 1 | Given that $DS = 5000H, CS = 0800H, SS = 0C20H, ES = F000H, BP = 0030H, SP = 0123H, SI = 0A12H, DI = ABCDH, BX = 5678H$; identify the addressing mode and determine the physical address resulting from the following:<br>$i)\ add\ ax,[si],\quad ii)\ mov\ al,[1000],\quad iii)\ mov\ [bx + si + 06],$<br>$iv)\ mov\ bx,[bp + 50], v)\ mul\ word\ ptr[BX]$ | [10] | CO2 | L3 |
| 2 | Discuss the addressing modes of 8086 in detail. | [10] | CO2 | L2 |
| 3 | Write an ALP to find the factorial of any number between 0H to 8H. | [10] | CO1 | L3 |
| 4 (a) | Discuss the different instruction formats of 8086. Also write the machine code for the following instructions if the first byte for MOVE instruction is "100010$DW$"<br>$i)\ MOV\ AL, num[SI], ii)\ MOV\ num[BX], DX$, where $num = 0100H$ | [07] | CO2 | L2 |
| (b) | Discuss the significance of instruction queue available in 8086. Justify why instruction queue is known as look-ahead feature of 8086? | [03] | CO1 | L2 |
| 5 | Explain the architecture of 8086 microprocessor with a neat block diagram. | [10] | CO2 | L3 |
| 6 | Write an ALP to compute the value of a function $Y = 3 * X + 5$. Where $X \rightarrow 0\ to\ 9$ and Adjust the value of $Y$ in unpacked BCD format. | [10] | CO2 | L3 |
| 7 | Write an ALP to find the number has an even or odd parity. If parity is even set DL to 00; else set DL to 01. Also to find the given number is an odd or even number. | [10] | CO2 | L3 |

| Course Outcomes | | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1: | Describe the architecture of 8086 processor | 3 | 3 | 2 | - | - | - | - | - | 1 | - | - | - |
| CO2: | Solve computational problems using instruction sets of 8086 microprocessor and 8087 coprocessor | 3 | 3 | 2 | - | - | - | - | - | 1 | - | - | - |
| CO3: | Analyze and optimize the execution time and memory requirement of a program. | 3 | 3 | 2 | - | - | - | - | - | 1 | - | - | - |
| CO4: | Apply software and hardware interrupts | 3 | 3 | 2 | - | - | - | - | - | 1 | - | - | - |
| CO5: | Illustrate the interfacing of different input and output peripherals withMicroprocessor and the communication through serial & parallel ports. | 3 | 3 | 2 | - | - | - | - | - | 1 | - | - | - |
| CO6: | Comparethe features of 80x86 family of Microprocessors | 3 | 3 | 2 | - | - | - | - | - | 1 | - | - | - |

| Cognitive level | KEYWORDS |
|---|---|
| L1 | List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc. |
| L2 | summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend |
| L3 | Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover. |
| L4 | Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer. |
| L5 | Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize. |

PO1 - *Engineering knowledge*; PO2 - *Problem analysis*; PO3 - *Design/development of solutions*; PO4 - *Conduct investigations of complex problems*; PO5 - *Modern tool usage*; PO6 - *The Engineer and society*; PO7- *Environment and sustainability*; PO8 – *Ethics*; PO9 - *Individual and team work*; PO10 - *Communication*; PO11 - *Project management and finance*; PO12 - *Life-long learning*

CMR

① Given that DS = 5000h, CS = 0800h, SS = 0C20h, ES = F000h
BP = 0030h, SP = 0123h, SI = 0A12h, DI = ABCDh, BX = 5678h
Identify the addressing mode and determine the physical address resulting from the following

(i) add ax,[SI]. (ii) mov al, [1000], (iii) mov [bx+SI+06], al
(iv) mov bx, [bp+50], (v) mul word ptr [BX].

— [10 marks]

→ Given : DS = 5000h, CS = 0800h, SS = 0C20h, ES = F000h
BP = 0030h, SP = 0123h, SI = 0A12h, DI = ABCDh, BX = 5678h

(i) add ax,[SI]

Addressing mode : Register or Indirect or Indexed addressing mode.

01 mark

Physical address calculation.

Segment address : [DS] = 5000h X 10h
= 50000h.

PA = [DS] x 10h + [SI] = 50000h + 0A12h.

PA = 50A12h.

— 01 mark

(ii) mov al, [1000h]

addressing mode: Direct addressing mode.

-01 mark

Physical address calculation:

[DS] x10h + Effective address [EA]

= 5000h x10h + 1000h

PA = 51000h.

-01 marks

(iii) mov [bx+si+06], al.

addressing mode: Based indexed with displacement addressing mode

01 mark

Physical address calculation:

[DS] x10h + [bx] + [SI] + Displacement =

= 5000h x10h + 5678h + 0A12h + 0006h

PA = 56090h

-01 marks

(iv) mov bx, [bp+50]:

Addressing mode: Based relative addressing mode.

01 marks

Physical addressing mode calculation:-

PA = [SS] x 10h + [BP] + 50

= 0C20h x 10h + 0030h + 50h

PA = 0C280h.

— 01 marks

(v) mul word ptr [BX]

addressing mode: Based addressing mode.

01 mark

Physical addressing mode

PA = [DS] x 10h + [BX]

= [5000h] x 10h + [5678h]

= 55678h.

— 01 mark

CMR

(2) Discuss the addressing modes of 8086 in detail. — 10 marks

→ The way in which an operand is specified is called its addressing mode.

(a) Immediate addressing mode:

Data.

8 or 16-bit ~~add~~ data is a part of the instruction

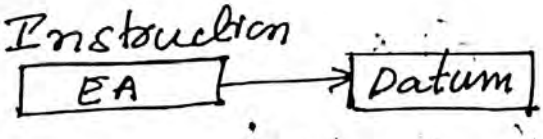
Instruction

| Datum |

Ex: mov AX, 0005h

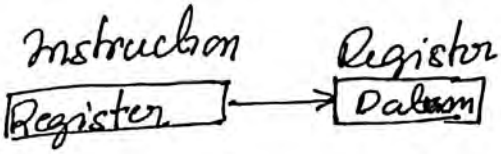
(b) Direct:

16-bit address of the datum is part of the instruction.




Ex: mov AX, [5000h]

(c) Register: Datum is in register as specified by the instruction




Ex: MOV BX, AX.

## ④ Register Indirect:

Effective address of the datum is in base register BX or an index register

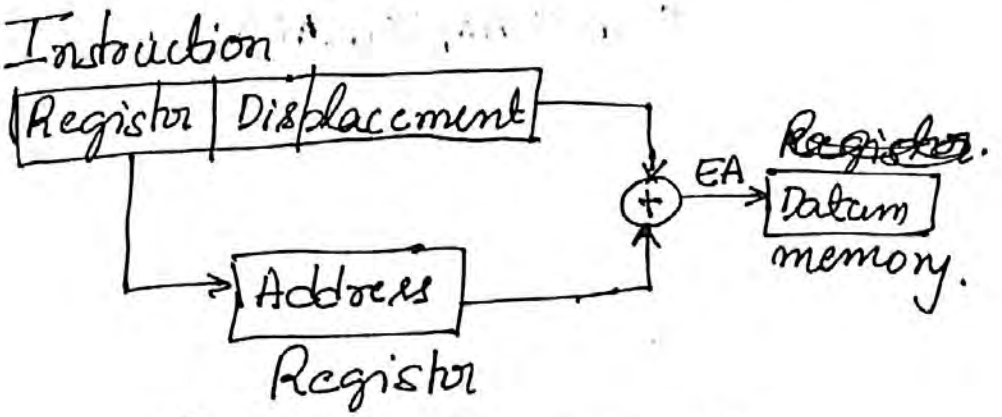
$$EA = \begin{Bmatrix} (BX) \\ (DI) \\ (SI) \end{Bmatrix}$$


-Ex: MOV AX, [BX]

## ⑤ Register Relative:

The effective address is sum of an 8-or 16-bit displacement and the contents of a base register or an index register.

$$EA = \begin{Bmatrix} (BX) \\ (BP) \\ (SI) \\ (DI) \end{Bmatrix} + \begin{matrix} \text{8-bit displacement} \\ \text{(sign extended)} \\ \text{16-bit displacement} \end{matrix}$$



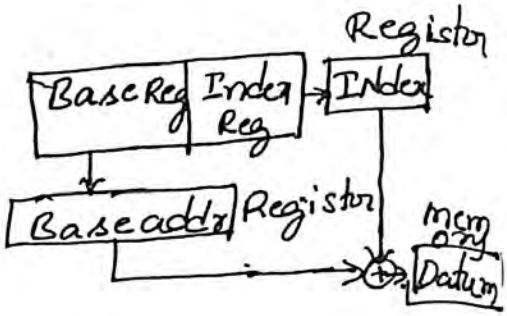

Ex: MOV AX, 50h [BX]

CMR

f) Based Indexed:

Effective address is the sum of base register and an index register.

$$EA = \begin{Bmatrix} (BX) \\ (BP) \end{Bmatrix} + \begin{Bmatrix} (SI) \\ (DI) \end{Bmatrix}.$$

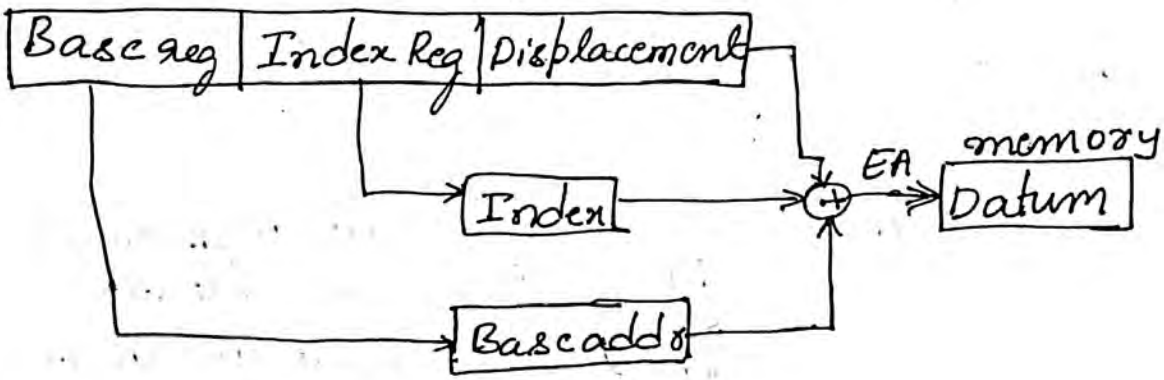



Ex: mov AX, [BX][SI]

g) Relative Based Indexed

EA is the sum of an 8-bit or 16-bit displacement and a based indexed address

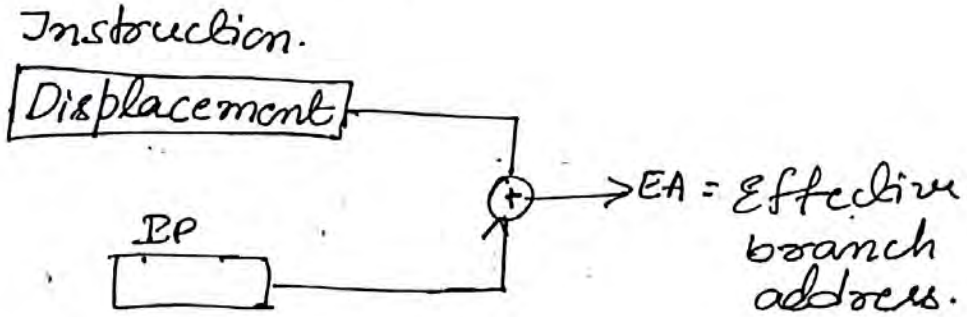
$$EA = \begin{Bmatrix} (BX) \\ (BP) \end{Bmatrix} + \begin{Bmatrix} (SI) \\ (DI) \end{Bmatrix} + \text{8-bit displa}$$




Ex: mov AX, 50h [BX][SI]

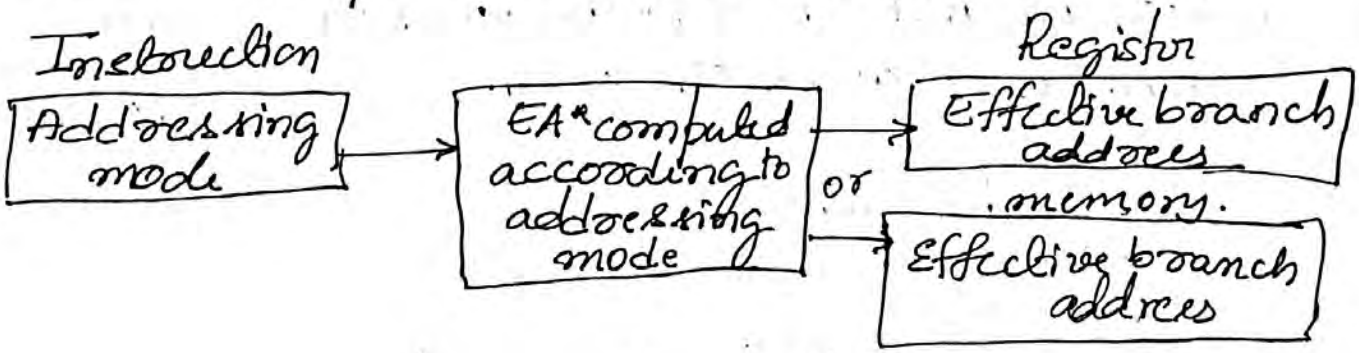
(h) Intrasegment direct:

Effective branch address is the sum of an 8- or 16-bit displacement and the current contents of IP. When the displacement is 8 bits long, referred as a short jump.

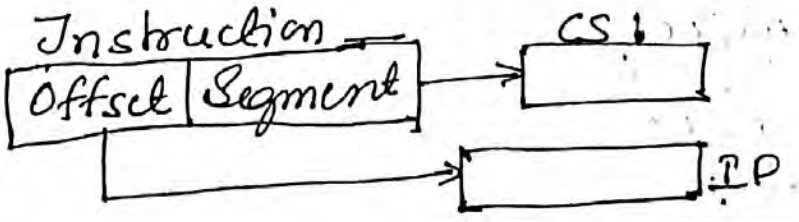



(i) Intrasegment Indirect.

Effective branch address is the contents of a register or memory location that is accessed using any of the above data-related addressing modes except the immediate mode.

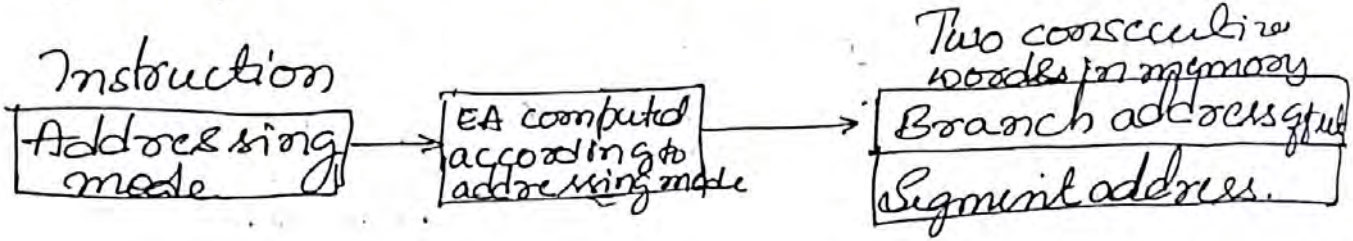



(j) Intersegment direct: Replaces the contents of IP with part of instruction and the contents of CS with another part of instruction.



## Intr Segment Indirect:

Replaces the contents of IP & CS with the contents of two consecutive words in memory that are referred referenced using any of the above data-related addressing modes (except the immediate and register modes.




10 addressing

10 X 1 = 10 marks

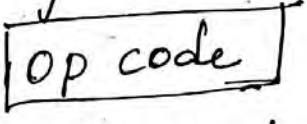
3) Write an ALP to find the factorial of any number between 0H to 8H. 10 Marks

```
.model
small
.data
    num dw 08h
    fact dw ?                                   03 Marks
.code
    start: mov ax, @data
           mov ds, ax
           mov ax,00h
            mov al, byte ptr num
           xor al, 0
           jnz skip
            mov fact, 01h
           jmp stop
     skip: mov al,01
     back: mul word ptr num
           dec num
           jnz back
           mov fact, ax
     stop: mov ax, 4c01h
           int 21h
    end start                                   07 Marks
```

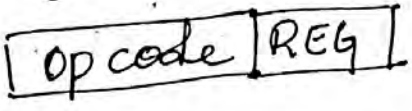
(4) (a)

Instruction size varies from one byte to six bytes. The opcode and addressing modes designations are in the first or second byte of an instruction.

(i) One byte instruction - implied operand.



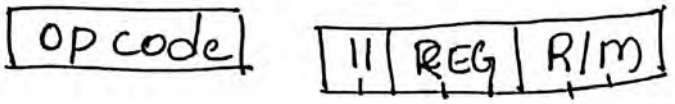

(ii) One byte instruction - register mode



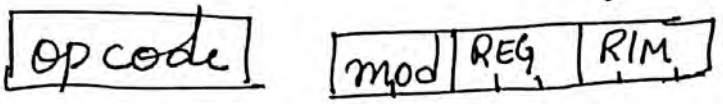

REG - Register
R/m - Register/memory
MOD - mode
Disp - Displacement
Data - Immediate data

CMR

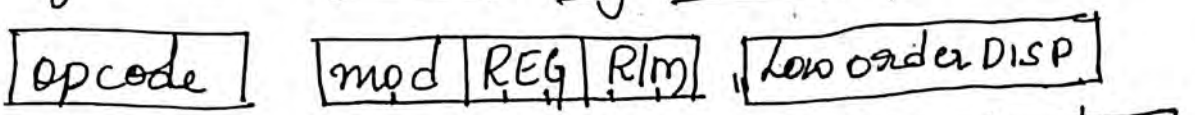
(iii) Register to Register




(iv) Register to/from memory with no displacement



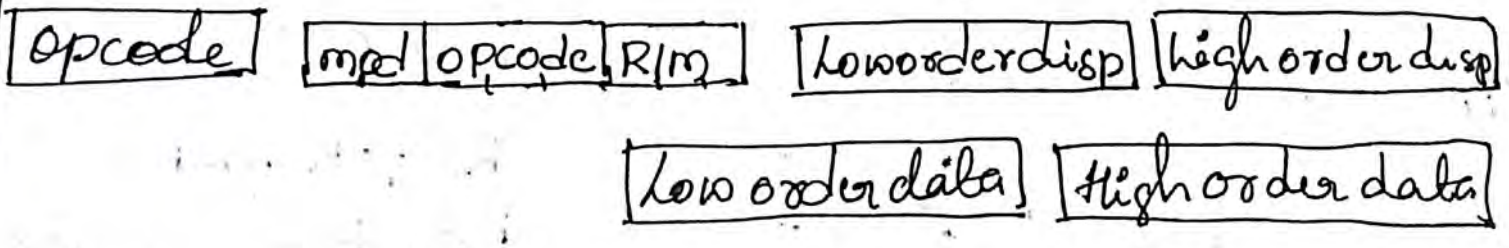

(v) Register to/from memory with displacement




if 16-bit displacement is used

(VI) Immediate operand to register.


(VII) Immediate operand to memory with 16-bit displacement



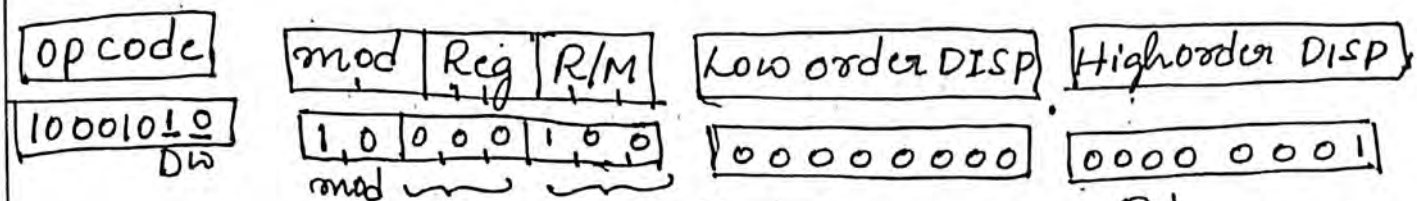

Any 6 X 0.5 marks each
= 3 marks

(i) mov AL, num[SI] where num = 0100h.

The addressing mode is Indexed relative or Register relative.

Instruction format is register from memory with displacement.

i.e.,




8A 84 00

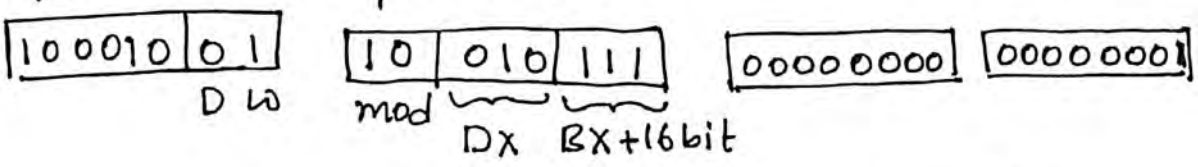
machine code is = 8A840001h

(ii) MOV num[BX], DX

Register Relative mode or.

Based Relative addressing mode.

Instruction format is register to memory with displacement.




89 97 00 01

machine code is = 89970001h -02 marks

CMR

4(b) Discuss the significance of instruction queue available in 8086. Justify why instruction queue is known as look-ahead feature of 8086? (6)

03 marks

→ Instruction queue is a 6 byte first-in first-out register of 8086. It is a 6 byte first-in first-out queue. This is continually being filled whenever the system bus is not needed for some other operation. Hence this instruction queue is called "look-ahead" feature of 8086. It increases the CPUs throughput because much of the time next instruction is already in the CPU when the present instruction completes its execution.

- 03 marks

5) Explain the architecture of 8086 microprocessor with a neat block diagram.

The block diagram of 8086 microprocessor architecture is as shown below:

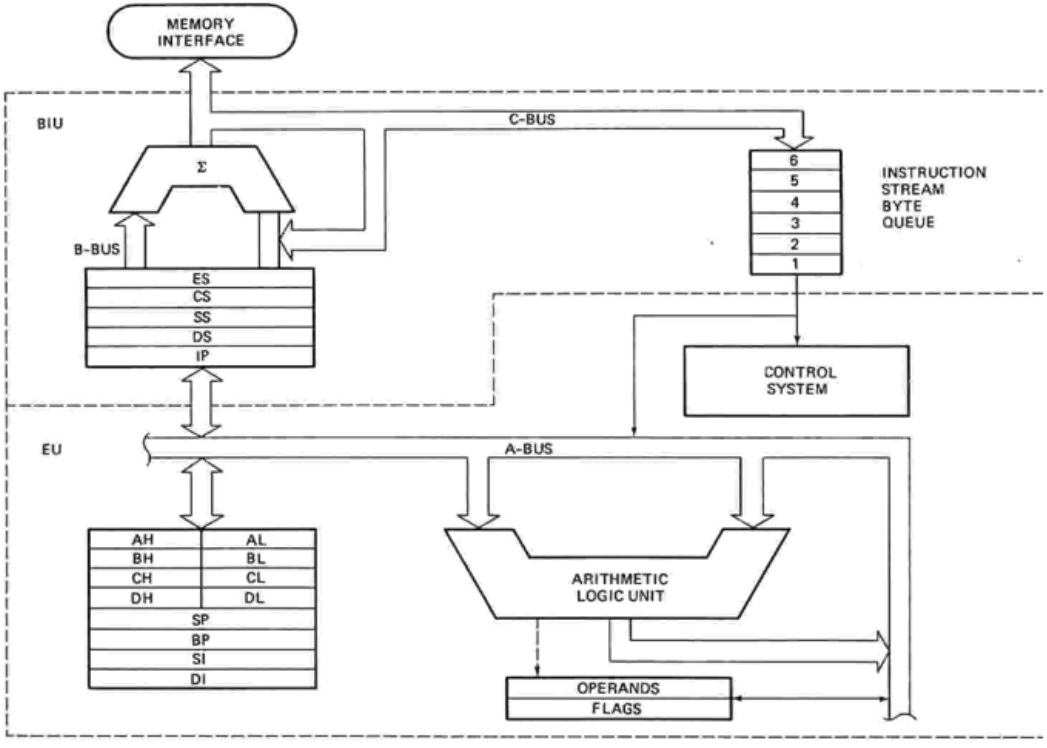



Fig: Architecture of 8086

The complete architecture of 8086 can be divided into two parts

(a) Bus interface unit (BIU)

(b) Execution unit. (EU).

Bus interface unit contains the circuit for physical address calculation and a preceding instruction byte queue (6 bytes long). The bus interface unit makes the system bus signals available for external interfacing of the devices. The Establishing of communication with the external devices and peripherals including memory. The complete physical address which is 20-bits long is generated using segment and offset registers, each 16-bits long. Instruction pointer register [IP] considered with code segment register [CS] to calculate the address of the instruction. Similarly stack pointer [SP] with stack segment register [SS], Destination index [DI] with extra segment and source index [SI], BX and/or [DI] or combination of all three register with Data segment register [DS] for ~~data~~ data.

All the register 16-bit long. Contents of segment registers is multiplied with 10h and is added with the contents of the registers as discussed above.

for example: CS = 1000h & IP = 5678h.

$$[CS] \times 10h + [IP] = [1000h] \times 10h + 5678h.$$
$$= 10000h + 5678h$$
$$= 15678h.$$

Instruction queue which is a 6 byte in size is used to prefetch the next instructions while the current instruction is being executed. While instruction is executed system bus is free, this time is utilized to fetch the next instruction. This could improve the performance throughput of 8086. while others exe instruction is fetching being fetched by BIU, EU can execute the decoded instruction.

Maximum one mega byte (1MB) of memory can be interfaced to 8086. 1MB memory is divided into different segments each of size 64kB.

The different segments are

1. Code Segment (CS)
2. Data Segment (DS)
3. Extra Segment (ES)
4. Stack Segment (SS)

Each segment starting address is stored in CS, DS, ES & SS registers. And it is converted into 20 bit address by multiplying with 10h.

Segments. can be overlapping and non overlapping.

Execution Unit (EU) consists of decoding circuit used to decode the instruction, along with the clocking and external control signals, control signals is generated. ALU is used to perform arithmetic and logical operations.

Register set is available in 8086. It consists of 4 16-bit general purpose registers namely. AX, BX, CX, DX. All these registers can also be programmed as 2 8-bit registers. For example AX can be used as AH and AL. 2 8-bit registers higher and lower bytes.

Similarily BX as BH & BL, CX as CH & CL and DX as DH & DL.

These 16 bit register can also perform some special functions. AX can be used as accumulator BX used to store base address hence it is called as base register & CX is used as counter and DX is used to store the address of the input address of input and output devices.

Along with general purpose registers, a set of special purpose registers are used such as SI, DI, BP & SP known as source index destination index, base pointer & stack pointer respectively.

Index registers are used to index the data
Stack pointer is used for stack memory operations
Base pointer is used to access any location of
stack memory.

A 16-bit register known as program status word used to store the status of the currently executing program.

Out of 16 bit only 9-bits are used others are unused. The organization of this register is as shown below

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| X | X | X | X | O | D | I | T | S | Z | X | A | X | P | X | C |

Carry flag (C) = Flag is set when there is a carry out of MSB or borrow into MSB in case of additions or subtraction.

Parity flag (P) = Flag is set to when lower byte of the result contains even number of 1's.

Auxillary carry flag (A): This is set if there is a carry from lower nibble to upper nibble.

Zero flag (Z): This is set if the result of the computation or comparison performed by the previous instruction

Sign flag (S): This flag is set, when the result of any computation is negative, Sign flag equals the MSB of the result.

CMR

Trap flag: If this flag is set, the processor enters the single step execution mode.

Interrupt flag (I): If this flag is set, the maskable interrupts are recognized by the CPU, otherwise they are masked.

Direction flag (D): This is used by string manipulation instructions. If this flag is '0', string is processed beginning from lower address to higher address. otherwise from higher to lower address.

Overflow flag (O): This is set, if the overflow occurs. If the result of a signed operation is large enough to be accommodated in a distination register. If carry from 6th bit to 7th bit is '1' and there is no carry from 7th bit is '0' or viceversa then overflow flag sets. otherwise overflow flag resets.

— 10 marks

3 (Diagram) + 3 (BIU) + 4 (EU)

6) Write an ALP to compute the value of a function $Y = 3 * X + 5$. Where $X \rightarrow 0\ to\ 9$ and Adjust the value of $Y$ in unpacked BCD format. 10 Marks

```
.model small
.data
m db 03h
x db 09h
c db 05h
y dw ?                                          03 Marks
.code
start: mov ax, @data
       mov ds, ax
       mov ax,00h
       mov al, m
       mul x
       aam
        mov bl, ah
        mov ah,00
       add al, c
       aaa
       add ah, bl
        mov byte ptr y, al
        mov byte ptr y+1, ah
       mov ax, 4c01h
       int 21h
end start                                       07 Marks
```

7) Write an ALP to find the number has an even or odd parity. If parity is even set DL to 00; else set DL to 01. Also to find the given number is an odd or even number. 10 Marks

```
.model small
.data
        num db 89h
        od db 0h
        ev db 0h                                    03 Marks
.code
start: mov ax, @data
        mov ds, ax
        mov ax,00h
         mov al, num
        xor al, 00
        jp skip
        mov dl, 01h
        jmp skip1
  skip: mov dl, 00h
skip1: ror al, 1
        jnc skip2
         mov od, 01h
         mov ev, 00h
        jmp skip3
skip2: mov od, 00h
         mov ev, 01h
 skip3: mov ax, 4c01h
         int 21h
         end start                                  07 Marks
```